SKRIPSI

HYPERHEURISTIC BERBASIS PARTICLE SWARM OPTIMIZATION UNTUK PERMASALAHAN PENJADWALAN JOB SHOP

Clara

NPM: 2013730004

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2017

UNDERGRADUATE THESIS

PARTICLE SWARM OPTIMIZATION BASED HYPERHEURISTIC FOR JOB SHOP SCHEDULING PROBLEM

Clara

NPM: 2013730004

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2017

LEMBAR PENGESAHAN

HYPERHEURISTIC BERBASIS PARTICLE SWARM OPTIMIZATION UNTUK PERMASALAHAN PENJADWALAN JOB SHOP

Clara

NPM: 2013730004

Bandung, 18 Desember 2017

Menyetujui,

Pembimbing

Dr.rer.nat. Cecilia Esti Nugraheni

Ketua Tim Penguji

Luciana Abednego, M.T.

Anggota Tim Penguji

Vania Natali, M.T.

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

HYPERHEURISTIC BERBASIS PARTICLE SWARM OPTIMIZATION UNTUK PERMASALAHAN PENJADWALAN JOB SHOP

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.
Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 18 Desember 2017

Clara
NPM: 2013730004

ABSTRAK

Skripsi ini menyelesaikan pengurutan *job shop scheduling problem* dengan menggunakan algoritma *Particle Swarm Optimization based Hyperheuristic*. *Job shop scheduling problem* adalah masalah optimasi di mana *job* diproses oleh sumber daya pada waktu-waktu tertentu. Urutan pengerjaan *job* pada penjadwalan *job shop* di setiap mesin dapat berbeda-beda. Banyak metode yang dapat digunakan dalam menyelesaikan penjadwalan *job shop*, salah satunya adalah dengan menggunakan *priority dispatch rule*. *Priority dispatch rule* dapat digunakan sebagai *low-level heuristic* atau yang biasa disebut *heuristic*. *Heuristic* adalah pendekatan untuk pemecahan masalah yang menggunakan metode yang memenuhi tujuan tapi tidak dijamin optimal atau sempurna. *Hyperheuristic* adalah *heuristic* yang memilih atau menghasilkan *heuristic*. *Hyperheuristic* menggunakan *low-level heuristic* dengan memilih *low-level heuristic* mana yang harus dipertimbangkan dan diterapkan. *Hyperheuristic* bertujuan menemukan metode yang tepat dalam situasi tertentu. Maka dari itu, *hyperheuristic* tidak memecahkan masalah secara langsung. Pada *Particle Swarm Optimization based Hyperheuristic*, partikel dari *Particle Swarm Optimization* akan menyimpan sejumlah *low-level heuristic* sebagai lokasi yang multi-dimensi.

Tujuan skripsi ini adalah membuat perangkat lunak yang dapat membantu menghasilkan jadwal yang lebih efisien atau jadwal yang memiliki *makespan* terkecil. *Makespan* adalah total waktu yang dibutuhkan untuk menyelesaikan seluruh *job*. Perangkat lunak yang dibangun tersebut memiliki *input* berupa *file* bertipe *'.txt'* berisi informasi dari *job*. *Output* dari perangkat lunak tersebut adalah jadwal tercepat yang didapat setelah melakukan sejumlah iterasi. Solusi dari *Particle Swarm Optimization* ini berupa himpunan dari beberapa bilangan bulat yang menandakan *low-level heuristic* yang akan digunakan untuk mengurutkan *job* di setiap mesin. *Priority dispatch rule* yang digunakan sebagai *low-level heuristic* pada skripsi ini adalah *first come first served*, *last in first out*, *shortest processing time*, *longest processing time*, *earliest due date*, *latest due date*, dan *random*.

Pengujian skripsi ini dilakukan terhadap 6 kasus permasalahan. Data yang digunakan adalah 3 kasus permasalahan sederhana dan *Taillard Benchmark*. *Taillard benchmark* adalah masalah yang belum terpecahkan dengan ukurannya sesuai dengan masalah industri yang dirancang Taillard untuk mengevaluasi metode yang sudah ada. Kasus permasalahan sederhana tersebut adalah 2 *job* 2 mesin, 3 *job* 4 mesin, dan 4 *job* 3 mesin. *Taillard benchmark* yang digunakan adalah 10 instansi dari kasus permasalahan 15 *job* 15 mesin. Solusi yang diperoleh dari 3 kasus permasalahan sederhana setelah beberapa iterasi adalah 15, 15, dan 20. Sedangkan dari pengujian *Taillard benchmark* dapat disimpulkan bahwa semakin besar permasalahan yang diuji, semakin kecil kemungkinan mendapat solusi yang optimal.

Kata-kata kunci: Penjadwalan *job shop, Particle Swarm Optimization, hyperheuristic, low-level heuristic, makespan, Taillard benchmark*

ABSTRACT

This thesis solves the job shop scheduling problem using the Particle Swarm Optimization based Hyperheuristic algorithm. Job shop scheduling problem is an optimization problem in which jobs are assigned to resources at particular times. In job shop scheduling problem, the order of processing job on each machine may vary. Many methods can be used to sort job shop, one of them is by using priority dispatch rule. Priority dispatch rule can be used as low-level heuristic or heuristic. Heuristic is an approach to problem solving that uses a purpose-filled method but is not guaranteed to be optimal or perfect. Hyperheuristic is heuristic that select or generate heuristics. Hyperheuristic uses low-level heuristic to find the right method. Hyperheuristic uses a low-level heuristic by selecting which low-level heuristic to consider and apply. Hyperheuristic aims to find the right method in a given situation. Therefore, hyperheuristic does not solve the problem directly. In Particle Swarm Optimization based Hyperheuristic, Particle Swarm Optimization particles will store a low-level heuristic as a multi-dimensional location.

The purpose of this thesis is to create software that can help generate a more efficient schedule or schedule that has the smallest makespan. Makespan is the total time it takes to complete all job. The built software has input in the form file of type '.txt' contains informations from jobs. Output of the software is the fastest schedule it gets after doing a number of iterations. The solution of Particle Swarm Optimization is a set of multiple integers indicating which low-level heuristic will be used to sort jobs on each machine. Priority dispatch rule used as low-level heuristic in this final project is first come first served, shortest processing time, longest processing time, earliest due date, latest due date, and random.

This final project will be conducted on 6 cases of problems. The data used are 3 cases of simple problems and Taillard benchmark. Taillad benchmark is an unsolved problem with their size in accordance with the industry issue that Taillard designed to evaluate existing methods. The simple problem cases are 2 job 2 machines, 3 job 4 machines, and 4 job 3 machines. Taillard benchmark used is 10 instances of the problem 15 job 15 machines. The solutions obtained from 3 simple problem cases after several iterations are 15, 15, and 20. While from the test Taillard benchmark it can be concluded that the bigger the problem being tested, the less likely it is to get the optimal solution.

Keywords: job shop scheduling, Particle Swarm Optimization, hyperheuristic, low-level heuristic, makespan, taillard benchmark

Dipersembahkan kepada diri sendiri serta keluarga yang selalu menyayangi dan mendukung.

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa yang telah memberikan kesehatan, kekuatan, keteguhan, anugerah, serta karunia yang luar biasa dan melimpah terutama dalam penyusunan serta menyelesaikan skripsi ini. skripsi ini diajukan untuk memenuhi salah satu syarat memperoleh gelar sarjana di Program Studi Informatika Universitas Parahyangan.

Penulis sadar penyusunan skripsi ini tidak terlepas dari bantuan, bimbingan, serta dukungan dari dukungan dan bantuan beberapa pihak yang diberikan secara langsung maupun tidak langsung. Oleh karena it, penuls ingin mengucapkan terima kasih kepada:

- Keluarga tercinta atas dukungan dan doanya agar penulis dapat menyelesaikan skripsi ini.
- Dr. rer. nat. Cecilia Esti Nugraheni, selaku dosen pembimbing penulis, yang selalu sabar menghadapi penulis dan memberi bimbingan, masukan, serta membantu penulis menyelesaikan skripsi ini.
- Jovan Gunawan atas doa, dukungan, semangat, serta kesabaran dalam menemani penulis mengerjakan skripsi ini.
- Teman-teman IT 2013
- Pihak-pihak lain yang tidak dapat disebutkan satu persatu.

Terakhir, penulis ingin meminta maaf jika ada kekurangan dan kesalahan dalam skripsi ini. Penulis harap skripsi ini bermanfaat bagi semua pihak yang memerlukan.

Bandung, Desember 2017

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Sekarang ini sudah banyak perusahaan yang didirikan, baik di bidang jasa maupun di bidang industri. Cepatnya perkembangan dan ketatnya persaingan di bidang industri membuat perusahaan-perusahaan tersebut menginginkan produk yang dihasilkan disukai atau sering digunakan oleh konsumen. Puas atau tidaknya seorang konsumen menjadi salah satu kunci kesuksesan dari suatu perusahaan. Hal ini disebabkan puas atau tidaknya seorang konsumen dapat meningkatkan atau menurunkan keuntungan yang didapat oleh perusahaan itu. Kepuasan konsumen dapat ditentukan oleh beberapa faktor, seperti misalnya kesesuaian produk dengan permintaan konsumen, harga yang tidak terlalu tinggi, mengikuti perkembangan zaman, kualitas produk, dan ketepatan waktu pada penyampaian produk ke tangan konsumen.

Masalah yang paling sering terjadi pada perusahaan industri adalah penyampaian produk ke tangan konsumen yang sering tidak tepat waktu, sehingga mengakibatkan keterlambatan pesanan. Hal ini disebabkan penjadwalan yang tidak benar atau tidak memiliki penjadwalan yang digunakan untuk mengatur proses produksi. Penjadwalan adalah kegiatan pengalokasian sumber-sumber atau mesin-mesin yang ada untuk menjalankan sekumpulan tugas dalam jangka waktu tertentu [1]. Penjadwalan produksi diupayakan untuk mendapatkan suatu penugasan *job* yang efektif pada setiap mesin agar tidak terjadi penumpukan *job* sehingga dapat mengurangi waktu menunggu (*delay*) untuk proses pengerjaan berikutnya.

Sistem informasi manajemen penjadwalan mesin merupakan upaya meningkatkan kinerja industri dalam mengelola, memproses data *resource* dan *constraint* yang tersedia. Pengelolaan tersebut bertujuan untuk menghasilkan informasi berkualitas dan efektif, sehingga mampu memberikan informasi dalam pengambilan keputusan jadwal mana yang terefektif jika digunakan.

Job shop scheduling problem adalah masalah optimasi dalam ilmu komputer dan riset operasi di mana pekerjaan yang ideal ditugaskan ke sumber daya pada waktu tertentu [2]. Pada *job shop scheduling problem*, yang harus dilakukan untuk memecahkan masalah adalah pengurutan pekerjaan dari jadwal pengerjaan produk yang tidak beraturan menjadi jadwal pengerjaan yang lebih ideal. Jadwal pengerjaan ideal yang dimaksud adalah jadwal yang memiliki *makespan* seminimal atau sekecil mungkin. *Makespan* adalah total waktu proses yang dibutuhkan untuk menyelesaikan suatu kumpulan *job* [3]. Dalam *job shop*, ada banyak *job* dengan berbagai tingkat prioritas dan *deadline* yang perlu dipenuhi yang harus diproses berbagai tipe mesin (penggilingan, pengeboran, dan lain-lain).

Sampai saat ini, sudah banyak penelitian yang dikembangkan untuk memecahkan *job shop scheduling problem*. Algoritma *Particle Swarm Optimization* adalah metode komputasi yang digunakan untuk menyelesaikan persoalan optimasi. *Particle Swarm Optimization* pertama kali ditemukan oleh Kennedy dan Eberhart pada tahun 1995 [4]. *Particle Swarm Opimtization* meniru atau dianalogikan dengan proses yang terjadi dalam kehidupan populasi burung, hewan herbivora, atau ikan. Salah satu proses yang dianalogikan adalah sekawanan burung yang sedang terbang untuk mencari sumber makanan. Dalam analogi ini, setiap individu burung dianggap sebagai partikel dalam *Particle Swarm Optimization*. Setiap partikel memiliki empat buah informasi, yaitu lokasi partikel tersebut saat ini, posisi terbaik yang pernah dikunjungi partikel tersebut, posisi terbaik yang pernah dikunjungi seluruh populasi, dan kecepatan

partikel tersebut. Partikel-partikel tersebut menggunakan ingatan yang dimiliki dirinya sendiri dan juga informasi yang diperoleh dari kawanannya secara keseluruhan untuk mencari solusi terbaik [5].

Heuristic atau yang disebut juga dengan heuristik adalah sebuah teknik yang mengembangkan efisiensi proses pencarian, namun dengan kemungkinan mengorbankan kelengkapan, optimalitas, akurasi, atau ketelitian. Menurut Schoenfeld, heuristik dapat juga disebut sebagai strategi umum yang membantu memecahkan masalah untuk mendekati dan memahami masalah serta menemukan solusi dari masalah [6].

Sedangkan, *hyperheuristic* adalah metode pencarian heuristik yang digunakan untuk memilih atau menghasilkan heuristik agar secara efisien dapat memecahkan masalah pencarian komputasi [7]. *Hyperheuristic* dapat didefinisikan sebagai heuristik yang memilih heuristik atau heuristik yang menghasilkan heuristik. Istilah *hyperheuristic* pertama kali digunakan pada tahun 2000, namun ide dari *hyperheuristic* sudah ada sejak tahun 1960. Ide utamanya adalah menemukan algoritma baru untuk menyelesaikan masalah kombinasi dengan menggunakan heuristik untuk memilih *low-level heuristics* mana yang sebaiknya digunakan untuk menyelesaikan masalah. Berbeda dengan beberapa aplikasi *metaheuristic* lain yang bekerja dengan mencari solusi, metode ini bekerja dengan mencari heuristik. *Metaheuristic* itu sendiri adalah metode lanjut berbasis heuristik untuk menyelesaikan persoalan optimisasi secara efisien [8]. *Hyperheuristic* memiliki 2 buah *layer* di dalam *framework*. Layar pertama adalah *hyperheuristic* itu sendiri. Sedangkan *layer* berikutnya adalah *layer* yang menyimpan sejumlah *low-level heuristic* yang dapat menyelesaikan masalah tertentu serta fungsi evaluasi. Kedua *layer* ini dipisahkan oleh masalah yang harus diselesaikan. *Layer* pertama perlu mengetahui jumlah *low-level heuristic* yang ada pada *layer* kedua. Selain itu, *hyperheuristic* memandu mencari solusi dengan mengatur strategi untuk memanggil dan mengevaluasi kinerja dari setiap *low-level heuristic*.

Penelitian ini membahas *hyperheuristic* berbasis *Particle Swarm Optimization* untuk *job shop scheduling problem*. *Hyperheuristic* berbasis *Particle Swarm Optimization* adalah *hyperheuristic* yang menggunakan *Particle Swarm Optimization* untuk menyelesaikan suatu masalah dan menghasilkan kualitas solusi yang lebih baik. Setiap partikel dalam populasi (*swarm*) beroperasi sama dengan *hyperheuristic*. Partikel-partikel tersebut bergerak pada ruang multidimensi, di mana setiap dimensinya merepresentasikan mesin pada penjadwalan. Maka dari itu, jumlah dimensi sama dengan jumlah mesin. Lokasi partikel pada suatu dimensi menunjukkan *heuristic* mana yang digunakan pada mesin. *Heuristic* yang digunakan adalah implementasi dari *priority dispatch rule*. *Priority dispatch rule* digunakan untuk mengurutkan *job* mana yang harus dikerjakan terlebih dahulu. Ada delapan *priority dispatch rule* yang umum digunakan, yaitu *First Come First Served*, *Last In First Served*, *Shortest Processing Time*, *Longest Processing Time*, *Earliest Due Date*, *Latest Due Date*, *Critical Ratio* dan *Random*. Kombinasi *heuristic* yang disimpan pada lokasi partikel membentuk jadwal *feasible* (layak) atau jadwal *infeasible* (tidak layak).

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas dalam penelitian ini :

1. Apa permasalahan utama dari *job shop scheduling problem*?
2. Bagaimana cara kerja *Particle Swarm Optimization* (PSO)?
3. Bagaimana cara kerja *hyperheuristic*?
4. Bagaimana mengimplementasikan atau menggunakan *hyperheuristic* berbasis *Particle Swarm Optimization* (PSO) untuk memecahkan masalah *job shop scheduling problem*?
5. Bagaimana performansi dari implementasi yang sudah dilakukan?

1.3 Tujuan

Tujuan yang ingin dicapai dalam penelitian ini :

1. Mengetahui masalah utama dari *job shop scheduling problem*
2. Mengetahui cara kerja dari *Particle Swarm Optimization* (PSO) dan *hyperheuristic*.
3. Membuat perangkat lunak yang dapat mengimplementasikan atau menggunakan *hyperheuristic* berbasis *Particle Swarm Optimization* (PSO) untuk memecahkan masalah *job shop scheduling problem*.
4. Menentukan bagus atau tidaknya performansi dari perangkat lunak yang sudah dibuat.

1.4 Batasan Masalah

Dalam penelitian ini ditetapkan batasan-batasan masalah sebagai berikut:

1. *Low-level heuristic* yang digunakan adalah *First Come First Served, Last In First Served, Shortest Processing Time, Longest Processing Time, Earliest Due Date, Latest Due Date,* dan *Random*.
2. Berdasarkan kriteria penjadwalan produksi, mesin yang digunakan adalah mesin jamak (*multi machine*) dengan pejadwalan statis dan informasi bersifat deterministik.

1.5 Metodologi

Metode penelitian yang akan digunakan dalam skripsi ini adalah:

1. Melakukan studi literatur mengenai penjadwalan mesin secara umum, *job shop scheduling problem, hyperheuristic,* dan *Particle Swarm Optimization*
2. Menganalisis atau mengeksplorasi *Particle Swarm Optimization based hyperheuristic* untuk *job shop scheduling problem* menggunakan bahasa *Java*.
3. Merancang dan mengimplementasi modul aplikasi yang sesuai dengan kebutuhan pengguna sehingga pengguna mendapatkan jadwal mesin yang memiliki *makespan* (hasil perbedaan waktu antara awal mula pekerjaan pertama diproses hingga akhir ketika seluruh pekerjaan selesai diproses) sekecil atau seminimal mungkin.
4. Menguji dan bereksperimen mengenai implementasi algoritma *hyperheuristic* berbasis *Particle Swarm Optimization*
5. Membuat dokumentasi skripsi.

1.6 Sistematika Pembahasan

Untuk penulisan skripsi ini akan dibagi dalam enam bagian sebagai berikut :

1. Bab 1 Pendahuluan

   Bab pendahuluan berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian dan sistematika pembahasan.

2. Bab 2 Landasan Teori

   Bab landasan teori berisi dasar-dasar teori yang akan digunakan dalam pembuatan aplikasi pembangkit jadwal mesin. Dasar-dasar teori yang akan digunakan diantaranya adalah penjadwalan mesin secara umum, *job shop scheduling problem, heuristic, hyperheuristic, Particle Swarm Optimization,* dan *hyperheuristic* berbasis *Particle Swarm Optimization*.

3. Bab 3 Analisis

   Bab analisis berisi contoh kasus, analisis perangkat lunak yang akan dibangun, *use case diagram*, skenario, dan diagram kelas sederhana.

4. Bab 4 Perancangan

   Bab perancangan berisi perancangan tampilan dan diagram kelas rinci.

5. Bab 5 Implementasi dan Pengujian

   Bab implementasi dan pengujian bersisi implementasi dan pengujian *hyperheuristic* berbasis *Particle Swarm Optimization* untuk *job shop scheduling problem*.

6. Bab 6 Kesimpulan dan Saran

   Bab kesimpulan dan saran berisi kesimpulan dari penelitian yang dilakukan dari awal hingga akhir serta saran untuk pengembangan selanjutnya.